AMENDMENTS TO THE SPECIFICATION

Please replace the paragraph at page 8, lines 3-15 with the following amended paragraph:

While the preferred embodiments described above can enhance the portability of a memory device, a problem can be encountered if the host device uses a file system that is different from the file system used to store the program code or the data in the memory device. For example, a host device configured with a DOS FAT file system may not be able to read a memory device operating with a write-once file system, such as the one described in U.S. patent application serial number 09/748,589, which is assigned to the assignee of the present invention and is hereby incorporated by reference. To avoid this difficulty, it is preferred that the memory device comprise a portion that is readable by the file system of the host device and that stores program code operative to enable the host device to read the portion of the memory device storing program code or data, as described in U.S. Patent No. 6,778,974 ~~patent application serial number _____ (Attorney Docket No. 10519/17; MD-44)~~, which is assigned to the assignee of the present invention and is hereby incorporated by reference.

Please replace the paragraphs at page 8, line 29 – page 10, line 7 with the following amended paragraphs:

The program code can be written in any suitable programming language. If the program code is written in a hardware-specific language, the program code can only be executed on host devices with a specific hardware platform. If the program code is written in a hardware-independent language, however, the program code can be executed on any host device having an

interpreter for translating the program code into machine code understandable by the hardware platform of the host device. Accordingly, program code written in a hardware-independent language increases the portability of the memory device, which may be particularly desired in transporting data across different types of consumer devices (*e.g.*, digital cameras, personal digital assistants (PDAs), projectors, etc.) One suitable hardware-independent language is Java®. The source code of a Java® program is compiled into an intermediate language called "bytecode." Bytecode is compiled for a theoretical machine, and a Java® interpreter (a Java® Virtual Machine) in a host device emulates that machine by converting the bytecode into machine code at runtime. Because bytecode is not dependent on any specific hardware platform, it will run in any host device with the Java® interpreter.

The following example illustrates the use of hardware-independent program code and will be discussed in conjunction with Figure 2. Figure 2 shows a solid-state memory device 20 and two host devices: a first host device 30 using a first hardware platform and a second host device 40 using a second hardware platform. Both the first and second host devices 30, 40 comprise a program code interpreter 35. In this example, the first host device 30 takes the form of a personal computer with an Intel Pentium processor using the Windows operating system, the second host device 40 takes the form of an overhead projector with a customized hardware platform, and the program code interpreter 35 takes the form of a Java® interpreter. The program code stored in the solid-state memory device 20 is a presentation program written in Java®. In this example, the user desires to create a presentation on his personal computer 30 and deliver the presentation using the overhead projector. The user first connects the solid-state memory device 20 with his personal computer 30. The Java® interpreter 35 converts the Java®-version of the presentation program into machine code appropriate for the computer's hardware

platform, and the code is executed. After the user generates his presentation with the presentation program, he saves the presentation on the solid-state memory device 20. The user then removes the solid-state memory device 20 and plugs it into the overhead projector 40. The Java® interpreter 35 of the overhead projector 40 converts the Java®-version of the presentation program into machine code for the projector 40, and the presentation program is executed on the projector 40. If an audience member asks for a copy of the slides used in the presentation, the user can plug the solid-state memory device 20 into the Java®-capable printer, and print the desired slides. As the preceding example illustrates, by using a hardware-independent language, program code stored on a solid-state memory device can be read across multiple platforms with virtually no effort by a user.